

Autonomous and Mobile Robotics

Prof. Giuseppe Oriolo

Motion Planning 3

Artificial Potential Fields

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI



SAPIENZA
UNIVERSITÀ DI ROMA

on-line planning

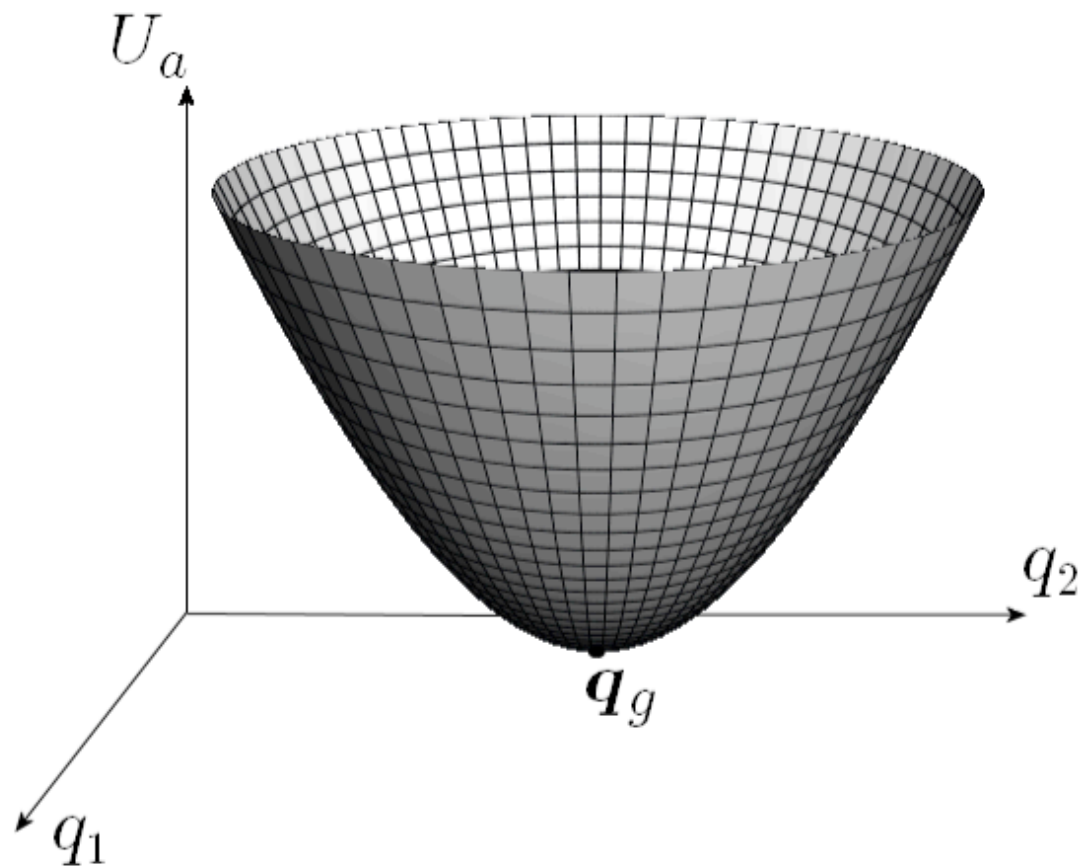
- autonomous robots must be able to plan **on line**, i.e, using **partial workspace information** collected during the motion via the robot sensors
- incremental workspace information may be integrated in a map and used in a **sense-plan-move** paradigm (**deliberative** navigation)
- alternatively, incremental workspace information may be used to plan motions following a memoryless **stimulus-response** paradigm (**reactive** navigation)

artificial potential fields

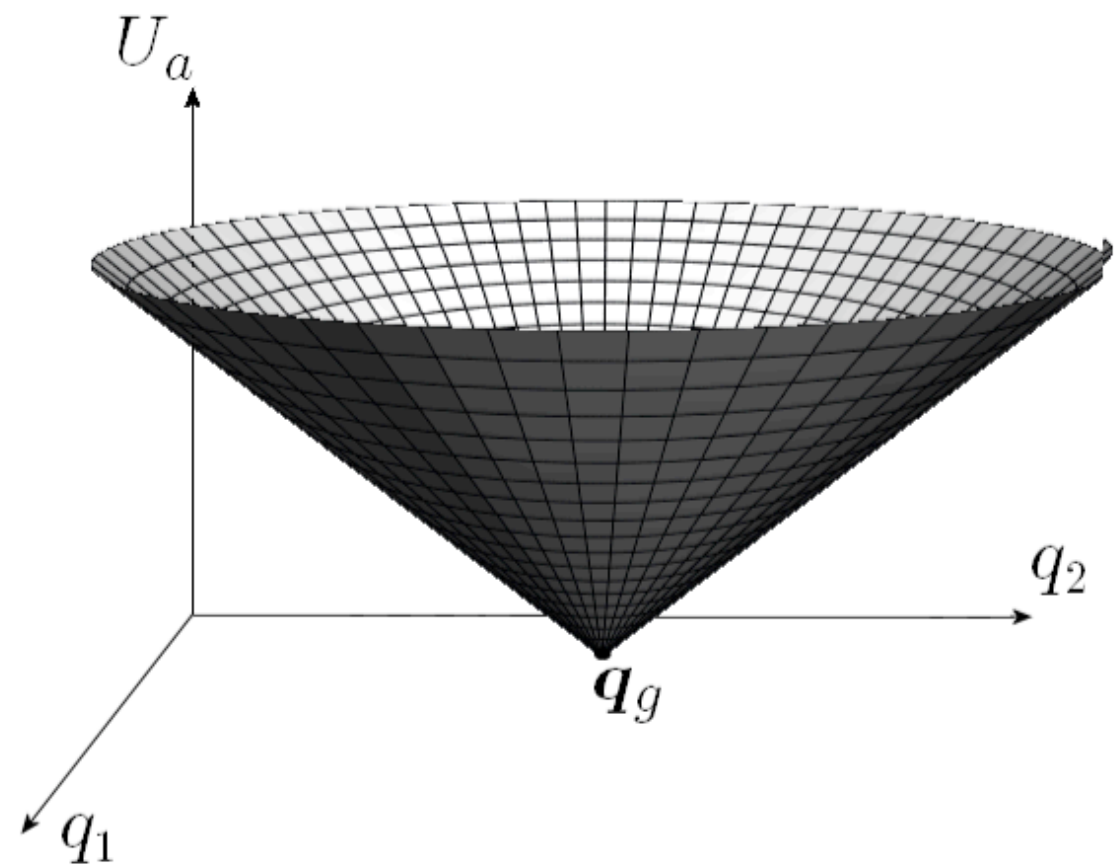
- **idea**: build potential fields in \mathcal{C} so that the point that represents the robot is **attracted** by the goal q_g and **repelled** by the \mathcal{C} -obstacle region \mathcal{CO}
- the total potential U is the sum of an **attractive** and a **repulsive** potential, whose negative gradient $-\nabla U(\mathbf{q})$ indicates the **most promising local direction** of motion
- the chosen **metric** in \mathcal{C} plays a role

attractive potential

- **objective**: to guide the robot to the goal q_g
- two possibilities; e.g., in $\mathcal{C} = \mathbb{R}^2$



paraboloidal



conical

- **paraboloidal**: let $e = q_g - q$ and choose $k_a > 0$

$$U_{a1}(q) = \frac{1}{2} k_a e^T(q) e(q) = \frac{1}{2} k_a \|e(q)\|^2$$

- the resulting attractive force is **linear** in e

$$f_{a1}(q) = -\nabla U_{a1}(q) = k_a e(q)$$

- **conical**:

$$U_{a2}(q) = k_a \|e(q)\|$$

- the resulting attractive force is **constant**

$$f_{a2}(q) = -\nabla U_{a2}(q) = k_a \frac{e(q)}{\|e(q)\|}$$

- f_{a1} behaves better than f_{a2} in the vicinity of q_g but increases indefinitely with e
- a convenient solution is to **combine** the two profiles: **conical away** from q_g and **paraboloidal close** to q_g

$$U_a(\mathbf{q}) = \begin{cases} \frac{1}{2} k_a \|\mathbf{e}(\mathbf{q})\|^2 & \text{if } \|\mathbf{e}(\mathbf{q})\| \leq \rho \\ k_b \|\mathbf{e}(\mathbf{q})\| & \text{if } \|\mathbf{e}(\mathbf{q})\| > \rho \end{cases}$$

continuity of f_a at the transition requires

$$k_a \mathbf{e}(\mathbf{q}) = k_b \frac{\mathbf{e}(\mathbf{q})}{\|\mathbf{e}(\mathbf{q})\|} \quad \text{for } \|\mathbf{e}(\mathbf{q})\| = \rho$$

i.e., $k_b = \rho k_a$

repulsive potential

- **objective**: keep the robot away from \mathcal{CO}
- assume that \mathcal{CO} has been partitioned in advance in **convex components** \mathcal{CO}_i
- for **each** \mathcal{CO}_i define a repulsive **field**

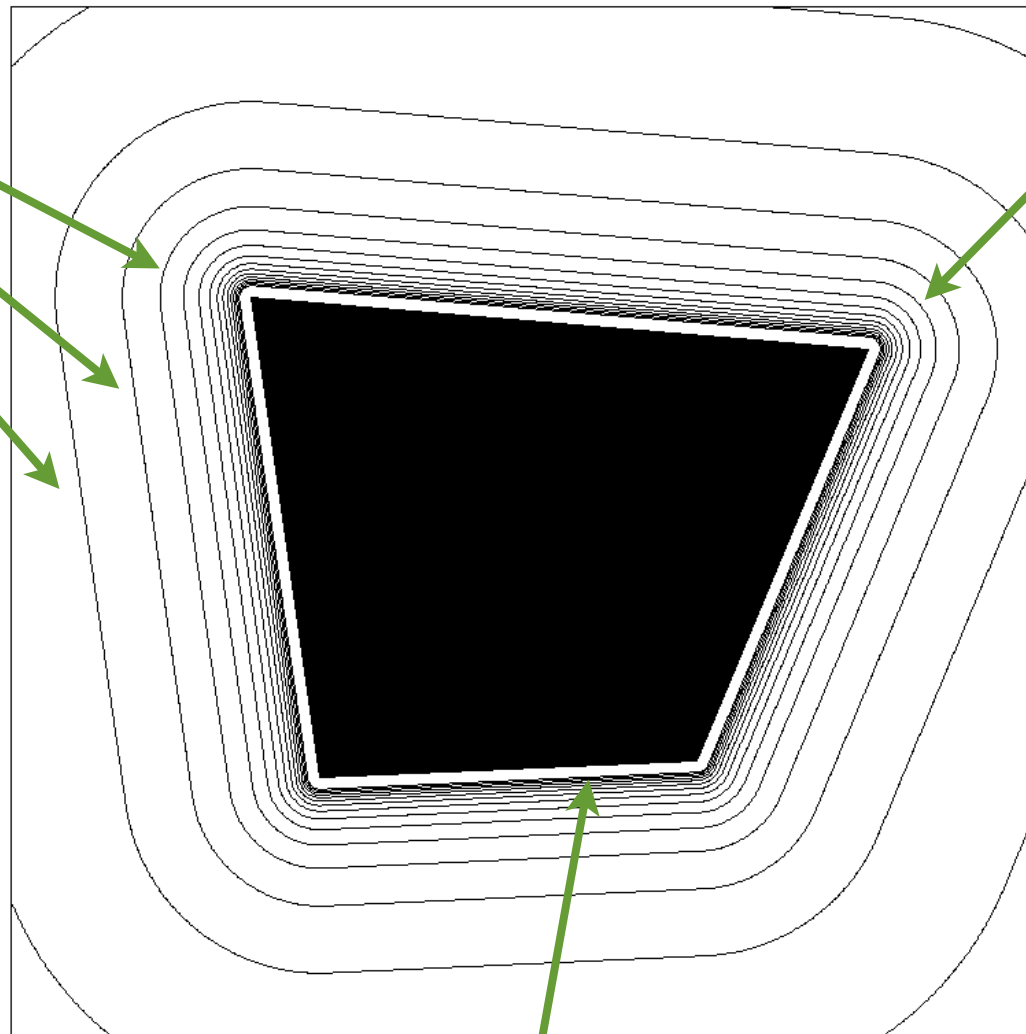
$$U_{r,i}(\mathbf{q}) = \begin{cases} \frac{k_{r,i}}{\gamma} \left(\frac{1}{\eta_i(\mathbf{q})} - \frac{1}{\eta_{0,i}} \right)^\gamma & \text{if } \eta_i(\mathbf{q}) \leq \eta_{0,i} \\ 0 & \text{if } \eta_i(\mathbf{q}) > \eta_{0,i} \end{cases}$$

where $k_{r,i} > 0$; $\gamma = 2, 3, \dots$; $\eta_{0,i}$ is the **range of influence** of \mathcal{CO}_i ; and

$$\eta_i(\mathbf{q}) = \min_{\mathbf{q}' \in \mathcal{CO}_i} \|\mathbf{q} - \mathbf{q}'\|$$

equipotential
contours

the higher γ ,
the steepest the slope



$U_{r,i}$ goes to ∞
at the boundary of \mathcal{CO}_i

- the resulting repulsive force is

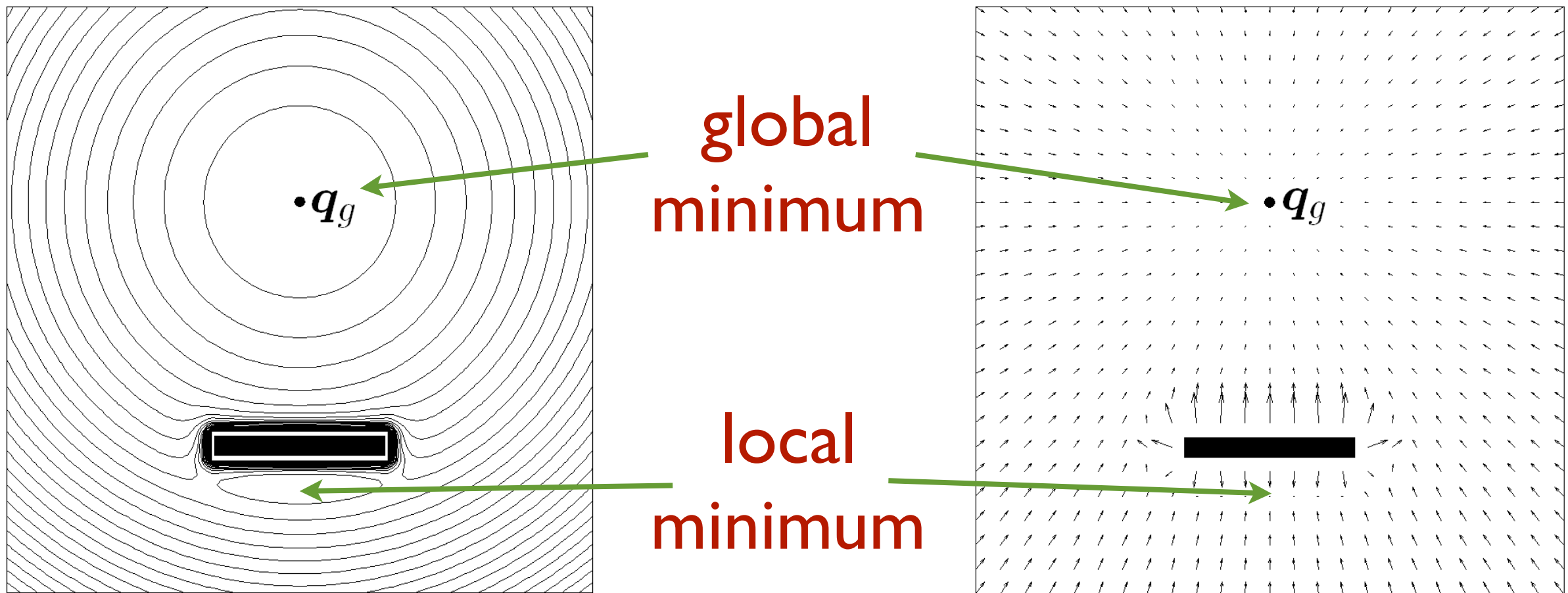
$$\mathbf{f}_{r,i}(\mathbf{q}) = -\nabla U_{r,i}(\mathbf{q}) = \begin{cases} \frac{k_{r,i}}{\eta_i^2(\mathbf{q})} \left(\frac{1}{\eta_i(\mathbf{q})} - \frac{1}{\eta_{0,i}} \right)^{\gamma-1} \nabla \eta_i(\mathbf{q}) & \text{if } \eta_i(\mathbf{q}) \leq \eta_{0,i} \\ 0 & \text{if } \eta_i(\mathbf{q}) > \eta_{0,i} \end{cases}$$

- $\mathbf{f}_{r,i}$ is **orthogonal to the equipotential contour** passing through \mathbf{q} and points away from the obstacle
- $\mathbf{f}_{r,i}$ is **continuous everywhere** thanks to the convex decomposition of \mathcal{CO}
- **aggregate** repulsive potential of \mathcal{CO}

$$U_r(\mathbf{q}) = \sum_{i=1}^p U_{r,i}(\mathbf{q})$$

total potential

- **superposition:** $U_t(\mathbf{q}) = U_a(\mathbf{q}) + U_r(\mathbf{q})$
- **force field:** $\mathbf{f}_t(\mathbf{q}) = -\nabla U_t(\mathbf{q}) = \mathbf{f}_a(\mathbf{q}) + \sum_{i=1}^p \mathbf{f}_{r,i}(\mathbf{q})$



planning techniques

- three techniques for planning on the basis of f_t
 1. consider f_t as generalized forces: $\tau = f_t(q)$
the effect on the robot is filtered by its dynamics
(generalized accelerations are scaled)
 2. consider f_t as generalized accelerations: $\ddot{q} = f_t(q)$
the effect on the robot is independent on its dynamics
(generalized forces are scaled)
 3. consider f_t as generalized velocities: $\dot{q} = f_t(q)$
the effect on the robot is independent on its dynamics
(generalized forces are scaled)

- **technique 1** generates **smoother** movements, while **technique 3** is quicker (irrespective of robot dynamics) to realize motion corrections; **technique 2** gives **intermediate** results
- strictly speaking, only **technique 3** guarantees (in the absence of local minima) **asymptotic stability** of q_g ; **velocity damping** is necessary to achieve the same with **techniques 1 and 2**

- off-line planning

paths in \mathcal{C} are generated by numerical integration of the dynamic model (if technique 1), of $\ddot{\mathbf{q}} = \mathbf{f}_t(\mathbf{q})$ (if technique 2), of $\dot{\mathbf{q}} = \mathbf{f}_t(\mathbf{q})$ (if technique 3)

the most popular choice is **3** and in particular

$$\mathbf{q}_{k+1} = \mathbf{q}_k + T \mathbf{f}_t(\mathbf{q}_k)$$

i.e., the algorithm of steepest descent

- on-line planning (is actually feedback!)

technique 1 directly provides control inputs, technique 2 too (via inverse dynamics), technique 3 provides reference velocities for low-level control loops

the most popular choice is **3**

local minima: a complication

- if a planned path enters the basin of attraction of a **local minimum** \mathbf{q}_m of U_t , it will reach \mathbf{q}_m and **stop** there, because $\mathbf{f}_t(\mathbf{q}_m) = -\nabla U_t(\mathbf{q}_m) = 0$; whereas saddle points are not an issue
- repulsive fields generally create local minima, hence **motion planning based on artificial potential fields is not complete** (the path may not reach \mathbf{q}_g even if a solution exists)
- **workarounds** exist but keep in mind that artificial potential fields are mainly used for **on-line** motion planning, where completeness may not be required

workaround no. 1: best-first algorithm

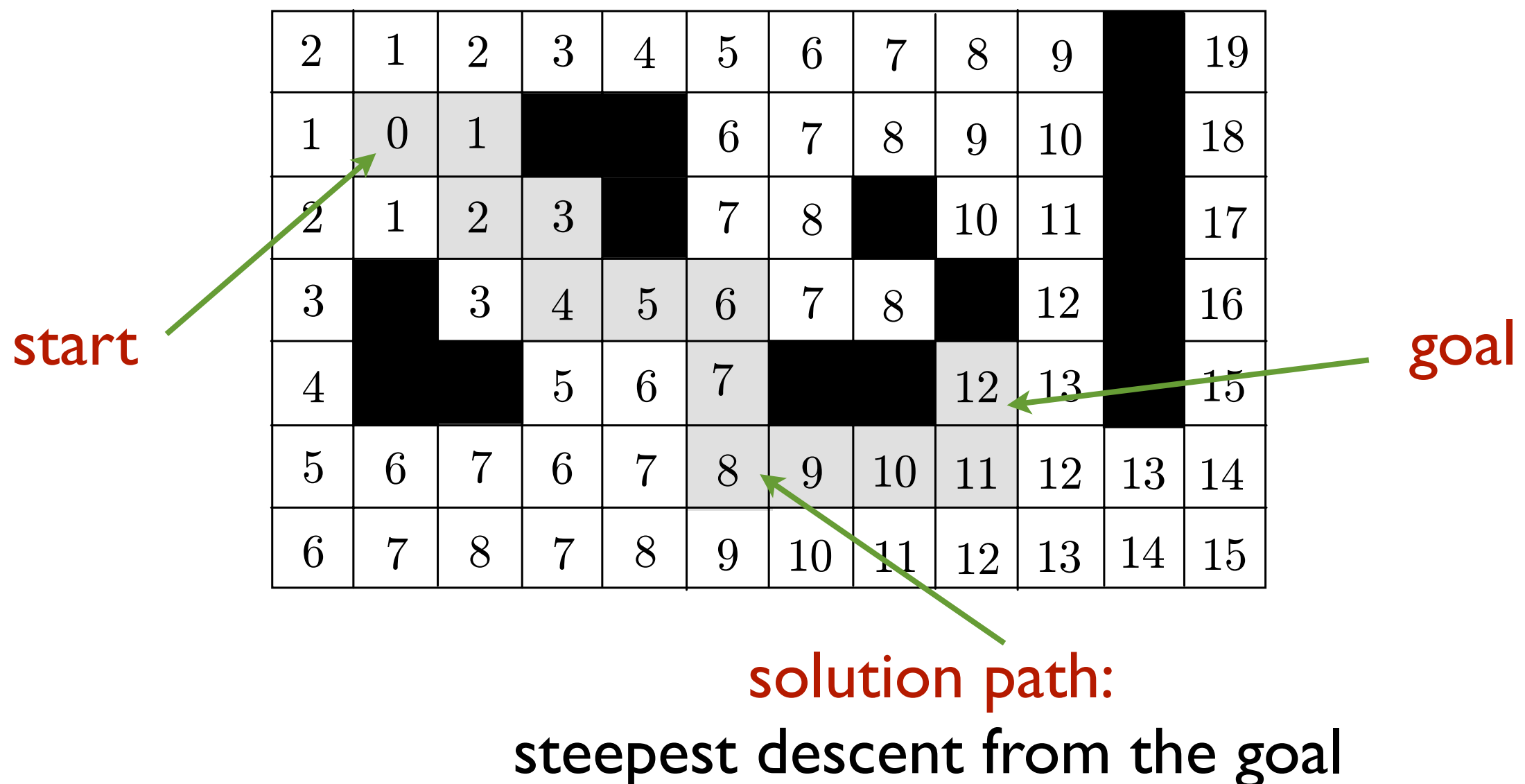
- build a **discretized representation** (by defect) of $\mathcal{C}_{\text{free}}$ using a regular grid, and associate to each free cell of the grid the value of U_t at its centroid
- build a tree T rooted at q_s : at each iteration, select the leaf of T with the **minimum** value of U_t and add as **children** its **adjacent free cells** that are not in T
- planning stops when q_g is reached (**success**) or no further cells can be added to T (**failure**)
- in case of success, build a solution path by **tracing back** the arcs from q_g to q_s

- best-first evolves as a **grid-discretized version of steepest descent** until a local minimum is met
- at a local minimum, best-first will “**fill**” its basin of attraction until it **finds a way out**
- the best-first algorithm is **resolution complete**
- its complexity is **exponential** in the dimension of \mathcal{C} , hence it is only applicable in low-dimensional spaces
- efficiency improves if random walks are alternated with basin-filling iterations (**randomized best-first**)

workaround no. 2: navigation functions

- paths generated by the best-first algorithm are **not efficient** (local minima are not avoided)
- a different approach: build **navigation functions**, i.e., potentials **without** local minima
- if the \mathcal{C} -obstacles are **star-shaped**, one can map \mathcal{CO} to a collection of spheres via a **diffeomorphism**, build a potential in transformed space and map it back to \mathcal{C}
- another possibility is to define the potential as an **harmonic function** (solution of Laplace's equation)
- all these techniques require **complete knowledge** of the environment: only suitable for **off-line planning**

- easy to build: **numerical navigation function**
- with $\mathcal{C}_{\text{free}}$ represented as a gridmap, assign 0 to start cell, 1 to cells adjacent to the 0-cell, 2 to unvisited cells adjacent to 1-cells, ... (**wavefront expansion**)



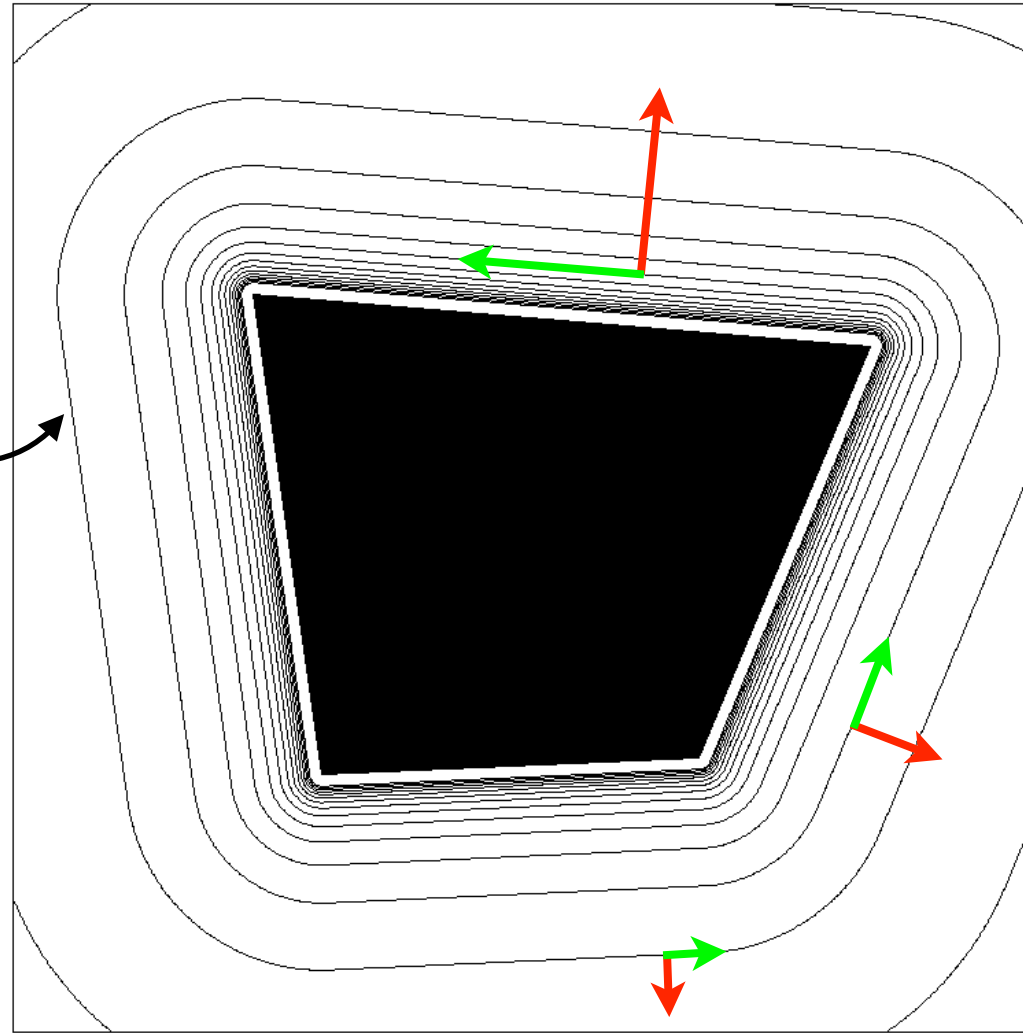
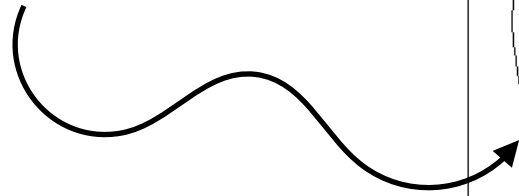
workaround no. 3: vortex fields

- an alternative to navigation functions in which one directly **assigns a force field** (rather than a potential)
- the idea is to replace the repulsive action (which is responsible for appearance of local minima) with an action **forcing the robot to go around the \mathcal{C} -obstacle**
- e.g., assume $\mathcal{C} = \mathbb{R}^2$ and define the **vortex field** for \mathcal{CO}_i as

$$\mathbf{f}_v = \pm \begin{pmatrix} \frac{\partial U_{r,i}}{\partial y} \\ -\frac{\partial U_{r,i}}{\partial x} \end{pmatrix}$$

i.e., a vector which is **tangent** (rather than normal) to the equipotential contours

equipotential
contours



f_r : repulsive

vs.

f_v : vortex

- the intensity of the two fields is the same, only the **direction** changes
- if \mathcal{CO}_i is convex, the vortex **sense** (CW or CCW) can be always chosen in such a way that the total field (attractive+vortex) has **no local minima**

- in particular, vortex sense (CW or CCW) should be chosen depending on the entrance point of the robot in the area of influence of the \mathcal{C} -obstacle
- vortex **relaxation** must be performed so as to avoid indefinite orbiting around the obstacle
- both these procedures can be easily performed at runtime based on **local sensor measurements**
- complete knowledge of the environment is not required: also **suitable for on-line planning**

artificial potentials for wheeled robots

- since WMRs are typically described by kinematic models, artificial potential fields for these robots are used at the **velocity** level
- however, robots subject to nonholonomic constraints **violate** the free-flying assumption
- as a consequence, the artificial force f_t **cannot** be directly imposed as a generalized velocity \dot{q}
- a possible approach: use f_t to generate a feasible \dot{q} via **pseudoinversion**

- the kinematic model of a WMR is expressed as

$$\dot{\mathbf{q}} = \mathbf{G}(\mathbf{q})\mathbf{u}$$

- since \mathbf{G} is $n \times m$, with $n > m$, it is in general impossible to compute \mathbf{u} so as to realize **exactly** a desired $\dot{\mathbf{q}}_{\text{des}}$
- however, a **least-squares** solution can be used

$$\mathbf{u} = \mathbf{G}^\dagger(\mathbf{q})\dot{\mathbf{q}}_{\text{des}} = \mathbf{G}^\dagger(\mathbf{q})\mathbf{f}_t$$

where

$$\mathbf{G}^\dagger(\mathbf{q}) = (\mathbf{G}^T(\mathbf{q})\mathbf{G}(\mathbf{q}))^{-1}\mathbf{G}^T(\mathbf{q})$$

- as an application, consider the case of a **unicycle** robot moving in a **planar** workspace; we have

$$\mathbf{G}(\mathbf{q}) = \begin{pmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{pmatrix} \Rightarrow \mathbf{G}^\dagger(\mathbf{q}) = \begin{pmatrix} \cos \theta & \sin \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

the **least-squares** solution corresponding to an artificial force $\mathbf{f}_t = (f_{t,x} \ f_{t,y} \ f_{t,\theta})^T$ is then

$$v = f_{t,x} \cos \theta + f_{t,y} \sin \theta$$

$$\omega = f_{t,\theta}$$

v may be interpreted as the orthogonal projection of the **cartesian** force $(f_{t,x} \ f_{t,y})^T$ on the sagittal axis

- assume that the unicycle robot has a circular shape, so that its **orientation is irrelevant** for collision
- one may build artificial potentials in a reduced $\mathcal{C}' = \mathbb{R}^2$ with \mathcal{C}' -obstacles simply obtained by **growing** the workspace obstacles by the robot radius
- in \mathcal{C}' , the attractive field pulls the robot towards (x_g, y_g) while repulsive fields push it away from \mathcal{C}' -obstacle boundaries (segments and arcs of circle)
- since \mathcal{C}' does not contain the orientation, the total field **will not include** a component $f_{t,\theta}$

- this degree of freedom can be exploited by letting

$$\omega = \dot{\theta} = k_{\theta} (\text{atan2}(f_{t,y}, f_{t,x}) - \theta)$$

whose rationale is to force the unicycle **to align with the total field**, so that \mathbf{f}_t can be better reproduced

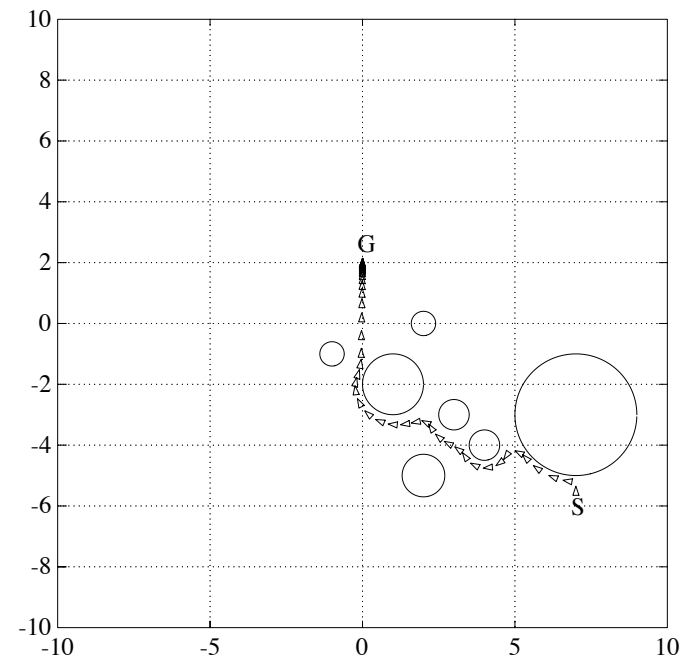
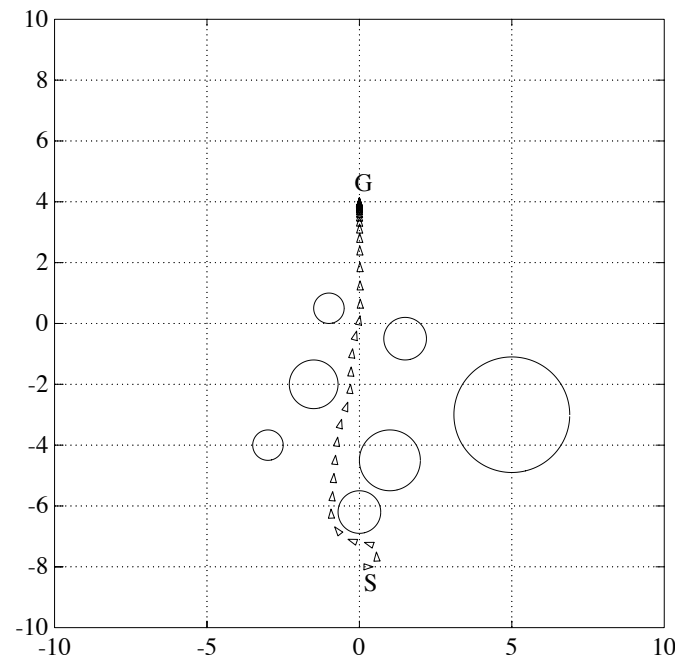
- overall, a **feedback control scheme** is obtained where v and ω are computed in real time from \mathbf{f}_t
- assume w.l.o.g. $(x_g, y_g) = (0, 0)$; **close to the goal**, where $\mathbf{f}_t = \mathbf{f}_a$, the controls become

$$v = -k_a (x \cos \theta + y \sin \theta)$$

$$\omega = k_{\theta} (\text{atan2}(-y, -x) - \theta)$$

i.e., a **cartesian regulator**! (see slides Wheeled Mobile Robots 5)

- results on unicycle (using vortex fields)



- can be applied to robots moving unicycle-like



motion planning for robot manipulators

- complexity of motion planning is **high**, because the configuration space has dimension typically ≥ 4
- try to **reduce** dimensionality: e.g., in 6-dof robots, replace the wrist with the total volume it can sweep (a conservative approximation)
- both the construction and the shape of \mathcal{CO} are complicated by the presence of **revolute** joints
- **off-line planning**: probabilistic methods are the best choice (although collision checking is heavy)
- **on-line planning**: adaptation of artificial potential fields

artificial potentials for robot manipulators

- to avoid the computation of \mathcal{CO} and the “curse of dimensionality”, the potential is built in \mathcal{W} (rather than in \mathcal{C}) and acts on a set of **control points** p_1, \dots, p_P distributed on the robot body
- in general, control points include **one point per link** (p_1, \dots, p_{P-1}) and the **end-effector** (to which the goal is typically assigned) as p_P
- the attractive potential U_a acts on p_P only, while the repulsive potential U_r acts on the whole set p_1, \dots, p_P ; hence, p_P is subject to the total $U_t = U_a + U_r$

- two techniques for planning with control points:

1. impose to the robot joints the **generalized forces** resulting from the combined action of force fields

$$\boldsymbol{\tau} = - \sum_{i=1}^{P-1} \mathbf{J}_i^T(\mathbf{q}) \nabla U_r(\mathbf{p}_i) - \mathbf{J}_P^T(\mathbf{q}) \nabla U_t(\mathbf{p}_P)$$

where $\mathbf{J}_i(\mathbf{q})$, $i=1, \dots, P$, is the **Jacobian** matrix of the direct kinematics function associated to $\mathbf{p}_i(\mathbf{q})$

2. use the above expression as **reference velocities** to be fed to the low-level control loops

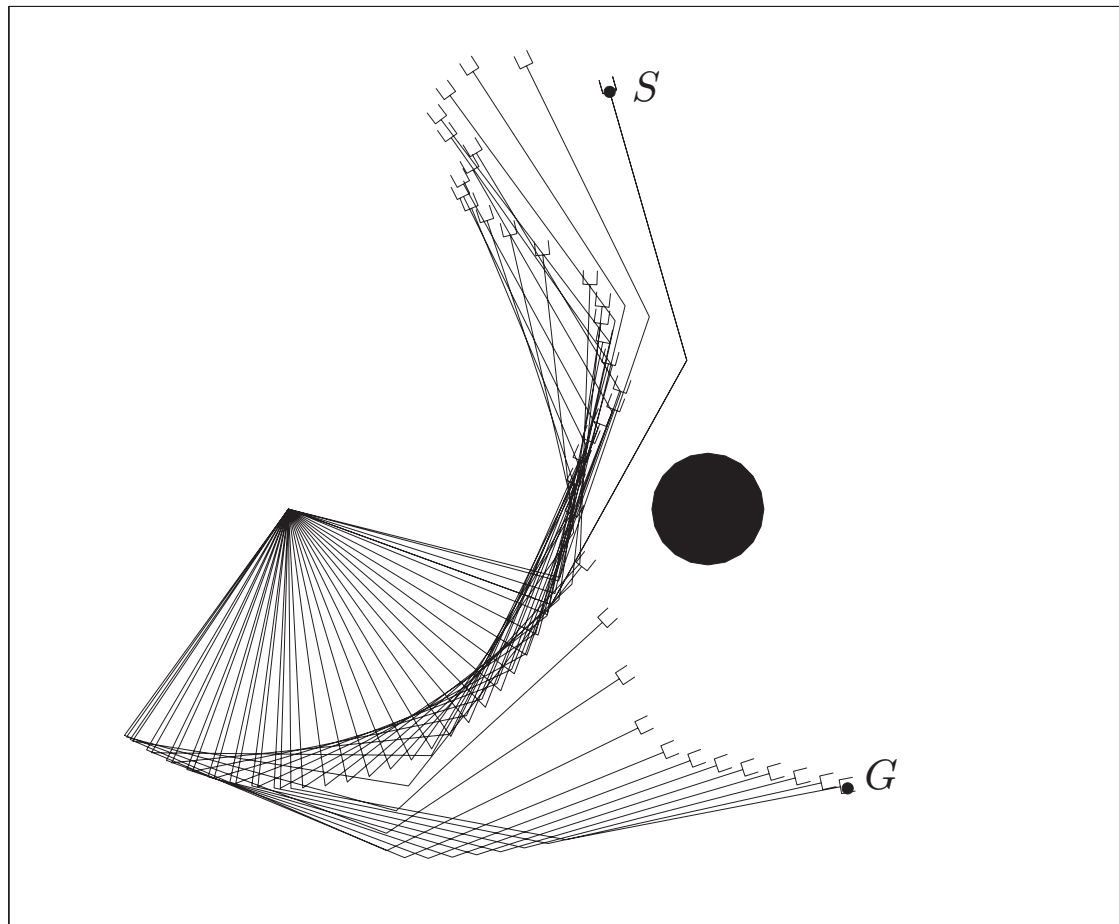
$$\dot{\mathbf{q}} = - \sum_{i=1}^{P-1} \mathbf{J}_i^T(\mathbf{q}) \nabla U_r(\mathbf{p}_i) - \mathbf{J}_P^T(\mathbf{q}) \nabla U_t(\mathbf{p}_P)$$

- **technique 2** is actually a **gradient-based minimization** step in \mathcal{C} of a combined potential in \mathcal{W} ; in fact

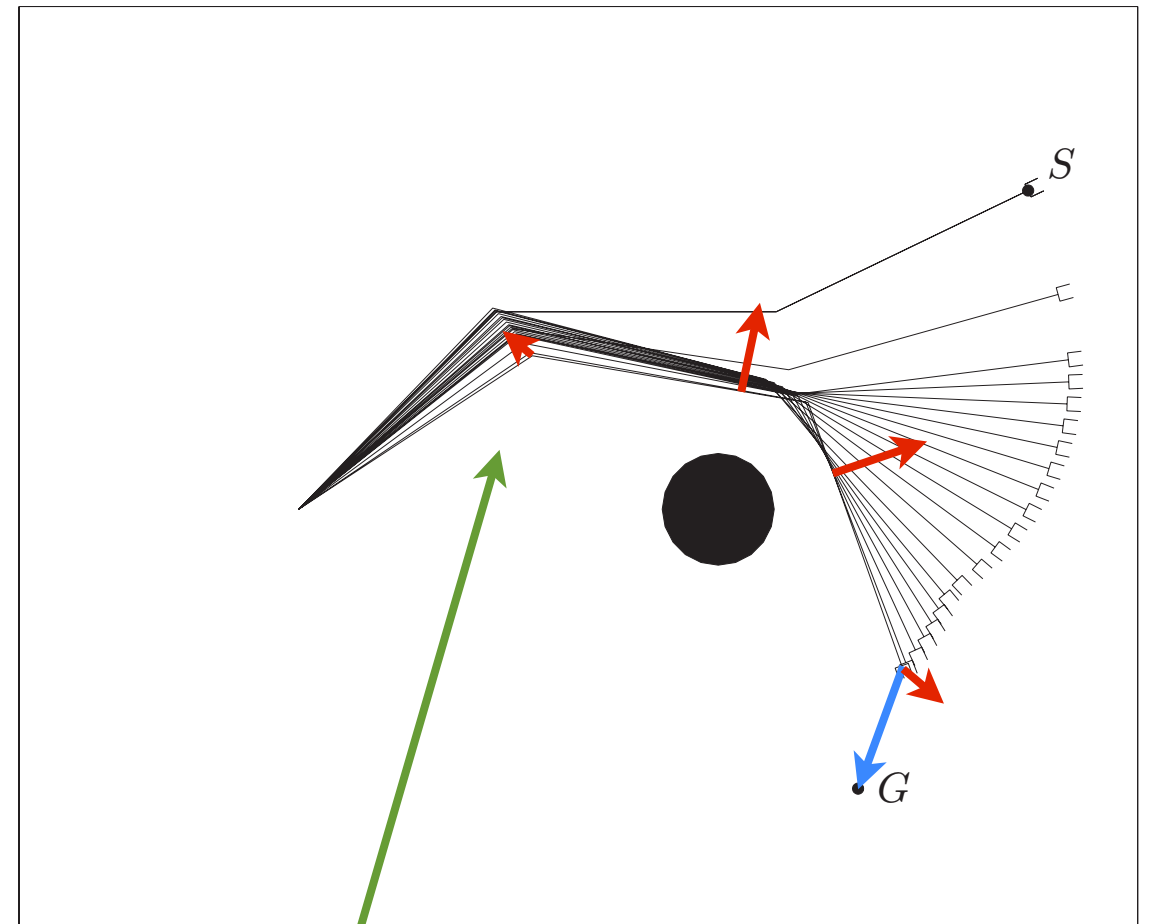
$$\nabla_{\mathbf{q}} U(\mathbf{p}_i) = \left(\frac{\partial U(\mathbf{p}_i(\mathbf{q}))}{\partial \mathbf{q}} \right)^T = \left(\frac{\partial U(\mathbf{p}_i)}{\partial \mathbf{p}_i} \frac{\partial \mathbf{p}_i}{\partial \mathbf{q}} \right)^T = \mathbf{J}_i^T(\mathbf{q}) \nabla U(\mathbf{p}_i)$$

- **technique 1** generates **smoother** movements, while **technique 2** is quicker (irrespective of robot dynamics) to realize motion corrections
- both can **stop** at **force equilibria**, where the various forces balance each other even if the total potential U_t is not at a local minimum; hence, this method should be used in conjunction with a **best-first algorithm**

success
(with **vortex** field
and **folding** heuristic for sense)



failure
(with **repulsive** field)



a force equilibrium
between **attractive** and **repulsive** forces